Noname manuscript No.
(will be inserted by the editor)

Certificates of infeasibility via nonsmooth optimization

Hannes Fendl · Arnold Neumaier · Hermann Schichl

the date of receipt and acceptance should be inserted later

Abstract An important aspect in the solution process of constraint satisfaction problems is to identify exclusion boxes which are boxes that do not contain feasible points. This paper presents a certificate of infeasibility for finding such boxes by solving a linearly constrained nonsmooth optimization problem. Furthermore, the constructed certificate can be used to enlarge an exclusion box by solving a nonlinearly constrained nonsmooth optimization problem.

Keywords Global optimization, nonsmooth optimization, certificate of infeasibility

Mathematics Subject Classification (2000) 90C26, 90C56, 90C57

1 Introduction

An important area of modern research is global optimization as it occurs very frequently in applications (extensive surveys on global optimization can be found in Neumaier [19], Floudas [7, 8], Hansen [10], and Kearfott [12]). A method for solving global optimization problems efficiently is by using a branch and bound algorithm (as, e.g., BARON by Sahinidis [22, 23], the COCONUT environment by Schichl [24, 25, 26], or LINGO by Schrage [27]), which divides the feasible set into smaller regions and then tries to exclude regions that cannot contain a global optimizer. Therefore, it is important to have tools which allow to identify such regions. In this paper we will present a method which is able to find such regions for a CSP (constraint satisfaction problem), i.e. for a global optimization problem with a constant objective function, by generalizing the approach from Fendl [4].

Certificate of infeasibility. For this purpose we consider the CSP

$$\begin{aligned} F(x) &\in \boldsymbol{F} \\ x &\in \boldsymbol{x} \end{aligned} \tag{1}$$

This research was supported by the Austrian Science Found (FWF) Grant Nr. P22239-N13.

Faculty of Mathematics, University of Vienna, Austria
Oskar-Morgenstern-Pl. 1, A-1090 Wien, Austria
E-mail: hermann.schichl@univie.ac.at,arnold.neumaier@univie.ac.at

with $F : \mathbb{R}^n \longrightarrow \mathbb{R}^m$, $\boldsymbol{x} \in \mathbb{IR}^n$, $\boldsymbol{F} \in \mathbb{IR}^m$, and we assume that a solver, which is able to solve a CSP, takes the box $\boldsymbol{u} := [\underline{u}, \overline{u}] \subseteq \boldsymbol{x}$ into consideration during the solution process. We constructed a certificate of infeasibility f, which is a nondifferentiable and nonconvex function in general, with the following property: If there exists a vector y with

$$f(y, \underline{u}, \overline{u}) < 0 \ , \tag{2}$$

then the CSP (1) has no feasible point in $\boldsymbol{u}$ and consequently this box can be excluded for the rest of the solution process. Therefore, a box $\boldsymbol{u}$ for which (2) holds is called an **exclusion box**.

Easy examples immediately show that there exist CSPs which have boxes that satisfy (2), so it is worth to pursue this approach further.

Exclusion boxes. The obvious way for finding an exclusion box for the CSP (1) is to minimize f

$$\min_{y} f(y, \underline{u}, \overline{u}) \tag{3}$$

and stop the minimization if a negative function value occurs. Since modern solvers offer many other possibilities for treating a box, we do not want to spend too much time for this minimization problem. Therefore, the idea is to let a nonsmooth solver only perform a few steps for solving (3).

To find at least an exclusion box $\boldsymbol{v} := [\underline{v}, \overline{v}] \subseteq \boldsymbol{u}$ with $\underline{v} + r \leq \overline{v}$, where $r \in (0, \overline{u} - \underline{u})$ is fixed, we can try to solve the linearly constrained problem

$$\begin{aligned} &\min_{y, \underline{v}, \overline{v}} f(y, \underline{v}, \overline{v}) \\ &\text{s.t. } [\underline{v} + r, \overline{v}] \subseteq \boldsymbol{u} \ . \end{aligned}$$

Another important aspect in this context is to enlarge an exclusion box $\boldsymbol{v}$ by solving

$$\begin{aligned} &\max_{y, \underline{v}, \overline{v}} \mu(\underline{v}, \overline{v}) \\ &\text{s.t. } f(y, \underline{v}, \overline{v}) \leq \delta \ , \ [\underline{v}, \overline{v}] \subseteq \boldsymbol{u} \ , \end{aligned} \tag{4}$$

where $\delta < 0$ is given and μ measures the magnitude of the box $\boldsymbol{v}$ (e.g., $\mu(\underline{v}, \overline{v}) := |\overline{v} - \underline{v}|_1$). Since only feasible points of (4) are useful for enlarging an exclusion box and we only want to perform a few steps of a nonsmooth solver as before, we expect benefits from a nonsmooth solver that only creates feasible iterates because then the current best point can always be used for our purpose. For proofs in explicit detail we refer the reader to Fendl [4, p. 147 ff, Chapter 5].

The paper is organized as follows: In Section 2 we first recall the basic facts of interval analysis which are necessary for introducing the certificate of infeasibility which is done afterwards. Then we discuss some important properties of the certificate and we explain in detail how the certificate is used for obtaining exclusion boxes in a CSP by applying a nonsmooth solver. In Section 3 we explain how we obtain a starting point for optimization problems to which we apply the nonsmooth solver.

Throughout the paper we use the following notation: We denote the non-negative real numbers by $\mathbb{R}_{\geq 0} := \{x \in \mathbb{R} : \ x \geq 0\}$ (and analogously for $\leq$ as well as $>$). Furthermore, we denote the p-norm of $x \in \mathbb{R}^n$ by $|x|_p$ for $p \in \{1, 2, \infty\}$.

2 Presentation of the application

After summarizing the most basic facts of interval analysis, we construct the certificate of infeasibility in this section. Furthermore, we discuss how a nonsmooth solver can use this certificate to obtain an exclusion box in a CSP.

2.1 Interval arithmetic

We recall some basic facts on interval arithmetic from, e.g., NEUMAIER [18]. We denote a box (also called interval vector) by $\boldsymbol{x} = [\underline{x}, \overline{x}]$ and the set of all boxes by $\mathbb{IR}^n := \{\boldsymbol{x} : \boldsymbol{x} = [\underline{x}, \overline{x}], \underline{x} \leq \overline{x}\}$. For $S \subseteq \mathbb{R}$ bounded, we denote the hull of S by $\Box S := [\inf S, \sup S]$. We extend the arithmetic operations and functions $\varphi : \mathbb{R} \longrightarrow \mathbb{R}$ to boxes by defining

$$\varphi(\boldsymbol{x}) := \Box\{\varphi(x) : x \in \boldsymbol{x}\} \ . \tag{5}$$

For every expression Φ of $\varphi : \mathbb{R}^n \longrightarrow \mathbb{R}$ which is a composition of arithmetic operations and elementary functions the fundamental theorem of interval arithmetic holds

$$[\inf_{x \in \boldsymbol{x}} \varphi(x), \sup_{x \in \boldsymbol{x}} \varphi(x)] \subseteq \Phi(\boldsymbol{x}) \ . \tag{6}$$

2.2 Certificate of infeasibility

Let $s : \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^q \times \mathbb{IR}^n \longrightarrow \mathbb{IR}$ be a function. We assume that $Z : \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^q \times \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}$

$$Z(y, z, w, \underline{x}, \overline{x}) := \sup s(y, z, w, \boldsymbol{x}) \ , \tag{7}$$

where $\boldsymbol{x} = [\underline{x}, \overline{x}] \in \mathbb{IR}^n$, satisfies

$$Z(y, z, w, \underline{x}, \overline{x}) \geq \sup_{x \in \boldsymbol{x}} y^T \big(F(x) - F(z)\big) \ . \tag{8}$$

Example 1 If we set $w = (R, S) \in \mathbb{R}^{n \times n}_{\text{triu}} \times \mathbb{R}^{n \times n}_{\text{striu}}$, where we denote the linear space of the upper resp. strictly upper triangular $n \times n$-matrices by $\mathbb{R}^{n \times n}_{\text{triu}} \cong \mathbb{R}^{n_1}$ with $n_1 := \frac{1}{2}n(n+1)$ resp. $\mathbb{R}^{n \times n}_{\text{striu}} \cong \mathbb{R}^{n_0}$ with $n_0 := \frac{1}{2}(n-1)n$, which implies $q = n_0 + n_1 = n^2$, and if we define

$$s_1(y, z, R, S, \boldsymbol{x}) := \Big(\sum_{k=1}^{m} y_k \mathfrak{F}_k[z, \boldsymbol{x}] + (\boldsymbol{x} - z)^T (R^T R + S^T - S) \Big) (\boldsymbol{x} - z) \tag{9}$$

then the corresponding Z satisfies (8) because: Due to the skew-symmetry of $S^T - S$, we have

$$y^T \big(F(x) - F(z)\big) + (x - z)^T (R^T R + S^T - S)(x - z) \geq y^T \big(F(x) - F(z)\big) \ . \tag{10}$$

for all $x \in \boldsymbol{x}$. Since the slope expansion

$$F_k(x) = F_k(z) + F_k[z, x](x - z) \ , \tag{11}$$

where the slope $F_k[z,x] \in \mathbb{R}^{1\times n}$, holds for all $x, z \in \mathbb{R}^n$ (cf., e.g., NEUMAIER [18]), we obtain for all $x \in \boldsymbol{x}$

$$y^T\big(F(x) - F(z)\big) + (x-z)^T(R^TR + S^T - S)(x-z) \subseteq s_1(y,z,R,S,\boldsymbol{x}) \tag{12}$$

due to (11), (5), (6), and (9). Now we obtain (8) due to (10), (12), and (7).

Proposition 1 *It holds for all $z \in \boldsymbol{x}$*

$$Z(y,z,w,\underline{x},\overline{x}) \geq 0\ . \tag{13}$$

Proof. (13) follows from (8) and the assumption that $z \in \boldsymbol{x}$. □

Definition 1 We define $Y : \mathbb{R}^m \times \mathbb{R}^n \longrightarrow \mathbb{R}$ and $f : \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^q \times \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}$ by

$$Y(y,z) := \inf y^T\big(\boldsymbol{F} - F(z)\big) \tag{14}$$

$$f(y,z,w,\underline{x},\overline{x}) := \frac{Z(y,z,w,\underline{x},\overline{x}) - \max\big(0, Y(y,z)\big)}{T(y,z,w,\underline{x},\overline{x})}\ , \tag{15}$$

where $T : \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^q \times \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}_{>0}$ is positive, continuous and differentiable almost everywhere.

Remark 1 f from (15) depends on $N = m + 3n + q$ variables and is not differentiable everywhere and not convex (in general).

Now we state the main theorem for our application.

Theorem 1 *If there exist $y \in \mathbb{R}^m$, $\underline{x} \leq z \leq \overline{x} \in \mathbb{R}^n$, and $w \in \mathbb{R}^q$ with $f(y,z,w,\underline{x},\overline{x}) < 0$, then for all $x \in \boldsymbol{x}$ there exists $k \in \{1,\ldots,m\}$ with $F_k(x) \notin \boldsymbol{F}_k$, i.e. there is no $x \in \boldsymbol{x}$ with $F(x) \in \boldsymbol{F}$, i.e. there is no feasible point.*

Proof. (by contradiction) Suppose that there exists $\hat{x} \in \boldsymbol{x} := [\underline{x}, \overline{x}]$ with $F(\hat{x}) \in \boldsymbol{F}$. By assumption there exist $y \in \mathbb{R}^m$ and $z \in \boldsymbol{x} \subseteq \mathbb{IR}^n$ with $f(y,z,w,\underline{x},\overline{x}) < 0$, which is equivalent to $Z(y,z,w,\underline{x},\overline{x}) < Y(y,z)$ due to (15) and (13). Since

$$Z(y,z,w,\underline{x},\overline{x}) \geq \sup_{x\in\boldsymbol{x}} y^T\big(F(x) - F(z)\big) \geq y^T\big(F(x) - F(z)\big)$$

for all $x \in \boldsymbol{x}$ due to (8) and

$$Y(y,z) \leq \inf_{\tilde{F}\in\boldsymbol{F}} y^T\big(\tilde{F} - F(z)\big) \leq y^T\big(\tilde{F} - F(z)\big)$$

for all $\tilde{F} \in \boldsymbol{F}$ due to (14) and (6), we obtain for all $x \in \boldsymbol{x}$ and for all $\tilde{F} \in \boldsymbol{F}$

$$y^T\big(F(x) - F(z)\big) \leq Z(y,z,w,\underline{x},\overline{x}) < Y(y,z) \leq y^T\big(\tilde{F} - F(z)\big)\ ,$$

which implies that we have $y^TF(x) < y^T\tilde{F}$ for all $x \in \boldsymbol{x}$ and for all $\tilde{F} \in \boldsymbol{F}$. Now, choosing $x = \hat{x} \in \boldsymbol{x}$ and $\tilde{F} = F(\hat{x}) \in \boldsymbol{F}$ in the last inequality yields a contradiction. □

The following proposition gives in particular a hint how the y-component of a starting point should be chosen (cf. (33)).

Proposition 2 *We have for all $y \in \mathbb{R}^m$ and $z \in \mathbb{R}^n$*

$$Y(y,z) = \sum_{k=1}^{m} \begin{cases} y_k\big(\underline{F}_k - F_k(z)\big) \text{ for } y_k \geq 0 \\ y_k\big(\overline{F}_k - F_k(z)\big) \text{ for } y_k < 0 \end{cases} . \tag{16}$$

Furthermore, let $I, J \subseteq \{1,\ldots,m\}$ satisfy $I \neq \emptyset \ \vee \ J \neq \emptyset$, $\underline{F}_i = -\infty \ \wedge \ y_i > 0$ for all $i \in I$ and $\overline{F}_j = \infty \ \wedge \ y_j < 0$ for all $j \in J$, then we have for all $z \in \mathbb{R}^n$

$$Y(y,z) = -\infty . \tag{17}$$

Proof. (16) holds because of (14). For obtaining (17), consider without loss of generality $\underline{F}_1 = -\infty \ \wedge \ y_1 > 0$ and $\overline{F}_2 = \infty \ \wedge \ y_2 < 0$, then the desired result follows from (16). □

2.3 Properties of the certificate for quadratic F

In this subsection we consider the special case of f with quadratic F, i.e.

$$F_k(x) = c_k^T x + x^T C_k x \tag{18}$$

with $c_k \in \mathbb{R}^n$ and $C_k \in \mathbb{R}^{n\times n}$ for $k = 1,\ldots,m$. Since

$$F_k(x) - F_k(z) = \big(c_k^T + x^T C_k + z^T C_k^T\big)(x - z)$$

for all $x, z \in \mathbb{R}^n$, the slope expansion from (11) holds with

$$F_k[z,x] = c_k^T + x^T C_k + z^T C_k^T . \tag{19}$$

Proposition 3 *Let F_k be quadratic and set*

$$\begin{aligned} C(y) := \sum_{k=1}^{m} C_k y_k, \ c(y,z) &:= \sum_{k=1}^{m} c_k y_k + \big(C(y) + C(y)^T\big)z, \ A(y,R,S) \\ &:= C(y) + R^T R + S^T - S . \end{aligned} \tag{20}$$

Then (8) is satisfied for

$$s_2(y,z,R,S,\boldsymbol{x}) := \big(c(y,z)^T + (\boldsymbol{x} - z)^T A(y,R,S)\big)(\boldsymbol{x} - z) . \tag{21}$$

Proof. Since $\sum_{k=1}^m y_k F_k[z,x] = c(y,z)^T + (x-z)^T C(y)$ due to (19) and (20), we obtain $\sum_{k=1}^m y_k F_k[z,x] + (x-z)^T(R^T R + S^T - S) = c(y,z)^T + (x-z)^T A(y,R,S)$ due to (20), which implies that s_2 from (21) has the same structure as s_1 from (9), and consequently we obtain that (8) holds for s_2, too. □

Proposition 4 *Let F_k be quadratic, then we have for all $p \in \mathbb{R}^m$ and $\alpha \in \mathbb{R}$*

$$C(y + \alpha p) = C(y) + \alpha C(p) , \quad c(y + \alpha p, z) = c(y,z) + \alpha c(p,z) \tag{22}$$

and furthermore we have for all $\kappa \geq 0$

$$A(\kappa^2 y, \kappa R, \kappa^2 S) = \kappa^2 A(y,R,S) . \tag{23}$$

Proof. (22) holds due to (20). (23) holds due to (20) and (22). □

Proposition 5 *Let F_k be quadratic. If the positive function $T : \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^{n_1} \times \mathbb{R}^{n_0} \times \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}_{>0}$ satisfies the (partial) homogeneity condition*

$$T(\kappa^2 y, z, \kappa R, \kappa^2 S, \underline{x}, \overline{x}) = \kappa^2 T(y, z, R, S, \underline{x}, \overline{x}) \tag{24}$$

for all $\kappa > 0$, $y \in \mathbb{R}^m$, $z \in [\underline{x}, \overline{x}] \in \mathbb{IR}^n$, $R \in \mathbb{R}^{n\times n}_{\text{triu}}$ and $S \in \mathbb{R}^{n\times n}_{\text{striu}}$, then the certificate f from (15) is (partially) homogeneous

$$f(\kappa^2 y, z, \kappa R, \kappa^2 S, \underline{x}, \overline{x}) = f(y, z, R, S, \underline{x}, \overline{x}) \ . \tag{25}$$

Proof. Since F_k is quadratic by assumption, the statements of Proposition 3 hold. By using (22) and (23) we calculate

$$c(\kappa^2 y, z)^T + (\boldsymbol{x} - z)^T A(\kappa^2 y, \kappa R, \kappa^2 S) = \kappa^2 \big(c(y, z)^T + (\boldsymbol{x} - z)^T A(y, R, S)\big) \tag{26}$$

and therefore $Z(\kappa^2 y, z, \kappa R, \kappa^2 S, \underline{x}, \overline{x}) = \kappa^2 Z(y, z, R, S, \underline{x}, \overline{x})$ follows due to (7), (21), and (26). Furthermore, we have $Y(\kappa^2 y, z) = \kappa^2 Y(y, z)$ due to (14) and, hence, we obtain $\max\big(0, Y(\kappa^2 y, z)\big) = \kappa^2 \max\big(0, Y(y, z)\big)$. Consequently, (15) and (24) imply (25). □

Remark 2 The intention of (25) is to reduce the scale dependence of the unbounded variables y, R and S of f. If we go through the proof of Proposition 5 again, we notice that we use the scaling property (23) of A for showing (26). From the proof of (23) we notice that this proof only holds, if y, R and S are treated as variables and none of them is treated as a constant (since factoring κ^2 out of a constant, yields an additional factor κ^{-2} to the constant). Nevertheless, if one of the variables y, R resp. S is treated as a constant and we set the corresponding value to $y = 0$, $R = 0$ resp. $S = 0$, then the proof still holds.

Example 2 Consider the variables R and S as constants and set $R = S = 0$. Then $T_1(y, z, R, S, \underline{x}, \overline{x}) := 1$ does not satisfy (24), while $T_2(y, z, R, S, \underline{x}, \overline{x}) := |y|_2$ does (cf. Remark 2). Note that T_2 violates the requirement of positivity, as demanded in Definition 1, for $y = 0$, and hence in this case f is only defined outside the zero set of T. Nevertheless, since the zero set of T_2 has measure zero, it is numerically very unlikely to end up at a point of this zero set and therefore we will also consider this choice of T due to the important fact of the reduction of the scale dependence of f as mentioned in Remark 2 (also cf. Subsection 2.4 (directly after optimization problem (27)) and Example 5).

Example 3 Choose $n = 2$, $m = 2$, $c_1 = \left(\begin{smallmatrix} 1 \\ -3 \end{smallmatrix}\right)$, $c_2 = \left(\begin{smallmatrix} 4 \\ 2 \end{smallmatrix}\right)$, $C_1 = \left(\begin{smallmatrix} 2 & 0 \\ 3 & 4 \end{smallmatrix}\right)$, $C_2 = \left(\begin{smallmatrix} -1 & 0 \\ -2 & 7 \end{smallmatrix}\right)$, which yields $F_1(x) = 2x_1^2 + x_1 + 3x_1x_2 - 3x_2 + 4x_2^2$ and $F_2(x) = -x_1^2 + 4x_1 - 2x_1x_2 + 2x_2 + 7x_2^2$ due to (18), $\boldsymbol{x} = [-3, 3] \times [-4, 4]$ and $\boldsymbol{F} = [-1, 7] \times [-2, 0]$, then we can illustrate certificate f from (15) with different T from Example 2 by the following plots

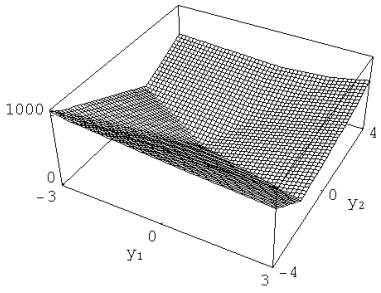


FIG. 1: $f(y_1, y_2)$ for T_1

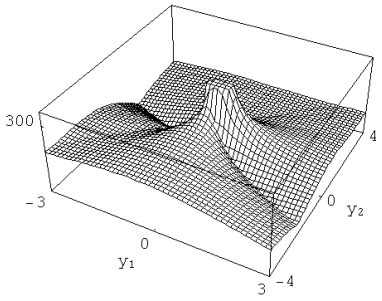


FIG. 2: $f(y_1, y_2)$ for T_2

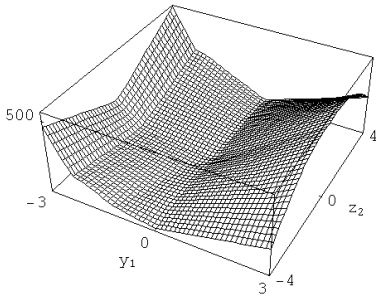


FIG. 3: $f(y_1, z_2)$ for T_1

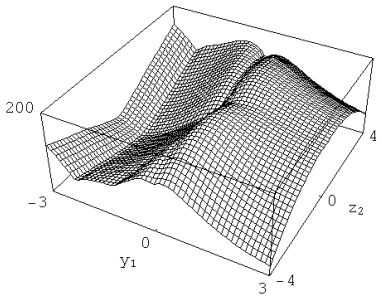


FIG. 4: $f(y_1, z_2)$ for T_2

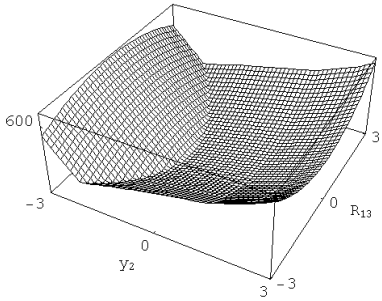


FIG. 5: $f(y_2, R_{13})$ for T_1

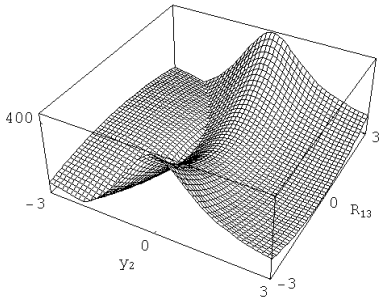


FIG. 6: $f(y_2, R_{13})$ for T_2

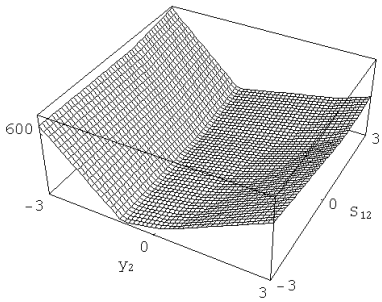


FIG. 7: $f(y_2, S_{12})$ for T_1

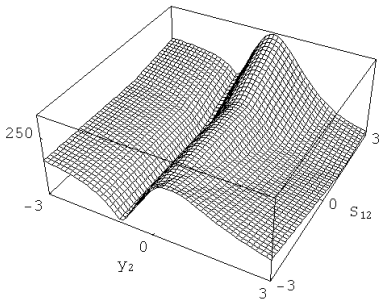


FIG. 8: $f(y_2, S_{12})$ for T_2

2.4 Exclusion boxes for constraint satisfaction problems

Now we explain in detail how to use Theorem 1 for finding exclusion boxes for the CSP (1). If we apply a solver for linearly constrained nonsmooth optimization (Note: The certificate f from (15) is not differentiable everywhere due to Remark 1) to

$$\begin{aligned} &\min f(y,z,w,u,v) \\ &\text{s.t. } u,v \in \boldsymbol{x} \ , \ u+r \leq v \ , \ z \in [u,v] \\ &\qquad y \in \mathbb{R}^m, z,u,v \in \mathbb{R}^n, w \in \mathbb{R}^q \end{aligned} \tag{27}$$

with a fixed $r \in [0, \overline{x} - \underline{x}]$ — although the convergence theory of many solvers (cf., e.g., the second order bundle algorithm by Fendl & Schichl [5, p. 7, 3.1 Theoretical basics]) requires that all occurring functions are defined on the whole $\mathbb{R}^N$, which might be violated for certain choices of T (cf. Example 2) — and if there occurs a function value smaller than zero (during the optimization process), then there is no feasible point in $[u,v]$ according to Theorem 1 and consequently we can reduce the box $\boldsymbol{x}$ to the set $\boldsymbol{x} \setminus [u,v]$ in the CSP (1).

If $[u,v] = \boldsymbol{x}$ (i.e. u and v are fixed and therefore no variables), then we can reduce the box $\boldsymbol{x}$ to the empty set, i.e. the reduction of a box to the empty set is equivalent to removing the box.

The constant r determines the size of the box $[u,v]$, which should be excluded: The closer r is to 0, the smaller the box $[u,v]$ can become (if $r = 0$, $[u,v]$ can become thin, what we want to prevent, since we want to remove a preferably large box $[u,v]$ out of $\boldsymbol{x}$, as then the remaining set $\boldsymbol{x} \setminus [u,v]$ is preferably small).

If during the optimization a point $z \in \boldsymbol{x}$ is found with $F(z) \in \boldsymbol{F}$, then we have found a feasible point and therefore we can stop the optimization, since then we cannot verify infeasibility for the box $\boldsymbol{x}$.

Remark 3 If $\boldsymbol{y} \subseteq \boldsymbol{x}$ and we remove $\boldsymbol{y}$ from $\boldsymbol{x}$, then the remaining set $\boldsymbol{x} \setminus \boldsymbol{y}$ is not closed. Nevertheless, if we just remove $\boldsymbol{y}^\circ \subset \boldsymbol{y}$, then the remaining set $\boldsymbol{x} \setminus \boldsymbol{y}^\circ \supset \boldsymbol{x} \setminus \boldsymbol{y}$ is closed (i.e. we remove a smaller box and therefore the remaining set is a bit larger, since it contains the boundary of $\boldsymbol{y}$). Furthermore, the set $\boldsymbol{x} \setminus \boldsymbol{y}$ can be represented as a union of at most $2n$ n-dimensional boxes, i.e. in particular the number of boxes obtained by this splitting process is linear in n.

We make the assumption that the certificate of infeasibility from (15) of the box $[\hat{u},\hat{v}]$ satisfies $f(\hat{y},\hat{z},\hat{w},\hat{u},\hat{v}) =: \hat{\delta} < 0$, i.e. $[\hat{u},\hat{v}]$ is an exclusion box according to Theorem 1. For $\delta \in [\hat{\delta}, 0)$ and a box $\boldsymbol{x}$ with $[\hat{u},\hat{v}] \subseteq \boldsymbol{x}$, we can try to apply a solver for nonlinearly constrained nonsmooth optimization to

$$\begin{aligned} &\min b(y,z,w,u,v) \\ &\text{s.t. } f(y,z,w,u,v) \leq \delta \\ &\qquad u,v \in \boldsymbol{x} \ , \ u \leq \hat{u} \ , \ \hat{v} \leq v \ , \ z \in [u,v] \\ &\qquad y \in \mathbb{R}^m, z,u,v \in \mathbb{R}^n, w \in \mathbb{R}^q \end{aligned} \tag{28}$$

to enlarge the exclusion box $[u,v]$ in $\boldsymbol{x}$, where $b : \mathbb{R}^N \longrightarrow \mathbb{R}$ is a measure for the box $[u,v]$ in the following sense: If $b : \mathbb{R}^N \longrightarrow \mathbb{R}_{\leq 0}$, then the following conditions must hold: If $[u,v]$ is small, then $b(.,u,v)$ is close to 0 and if $[u,v]$ is large, then $b(.,u,v)$ is negative and large. This means: The larger the box $[u,v]$ is, the more negative $b(.,u,v)$ must be. For examples of this type of box measure cf. (30). Alternatively, if $b : \mathbb{R}^N \longrightarrow \mathbb{R}_{\geq 0}$,

then the following condition must hold: If $[u,v] \subseteq \boldsymbol{x}$ is close to $\boldsymbol{x}$, then $b(.,u,v)$ is close to 0. For examples of this type of box measure cf. (31).

Remark 4 In opposite to (27), where the linear constraint $u + r \leq v$ occurs, we use in (28) the bound constraints $u \leq \hat{u}$ and $\hat{v} \leq v$.

Furthermore, we make the following two observations for the global optimization problem

$$\begin{aligned} &\min_x F_{\mathrm{obj}}(x) \\ &\text{s.t. } F_i(x) \in \boldsymbol{F}_i \quad \text{for all } i = 2,\ldots,m \\ &\qquad x \in \boldsymbol{x} \ , \end{aligned} \tag{29}$$

where $F_{\mathrm{obj}} : \mathbb{R}^n \longrightarrow \mathbb{R}$: First of all, the certificate f from (15) can be used for finding exclusion boxes in the global optimization problem (29) with an arbitrary objective function F_{obj}, since the certificate f only depends on the constraint data F, $\boldsymbol{F}$ and $\boldsymbol{x}$ (cf. the CSP (1)) and since a solution of an optimization problem is necessarily feasible. Secondly, we denote the current lowest known function value of the optimization problem (29) by $F_{\mathrm{obj}}^{\mathrm{cur}}$. Now, if we can find a box $[u,v] \subseteq x$ for which the certificate f from (15) with $\boldsymbol{F}_1 := [-\infty, F_{\mathrm{obj}}^{\mathrm{cur}}]$ has a negative value, then Theorem 1 implies that for all $x \in [u,v]$ there exists $k \in \{1,\ldots,m\}$ with $F_k(x) \notin \boldsymbol{F}_k$, which is equivalent that for all $x \in [u,v]$ we have $F_{\mathrm{obj}}^{\mathrm{cur}} < F_1(x)$ or there exists $i \in \{2,\ldots,m\}$ with $F_i(x) \notin \boldsymbol{F}_i$, i.e. any point in the box $[u,v]$ has an objective function value which is higher than the current lowest known function value $F_{\mathrm{obj}}^{\mathrm{cur}}$ or is infeasible. Consequently, the box $[u,v]$ cannot contain a feasible point with function value lower equal $F_{\mathrm{obj}}^{\mathrm{cur}}$, and hence the box $[u,v]$ cannot contain a global minimizer of the global optimization problem (29). Therefore we can exclude the box $[u,v]$ from further consideration.

Example 4 For measuring the box $[u,v]$ with $b : \mathbb{R}^N \longrightarrow \mathbb{R}_{\leq 0}$, we can use any negative p-norm with $p \in [1,\infty]$ as well as variants of them

$$\begin{aligned} &b_-^1(y,z,w,u,v) := -|v-u|_1 \ , \qquad b_-^2(y,z,w,u,v) := -\tfrac{1}{2}|v-u|_2^2 \ , \\ &b_-^\infty(y,z,w,u,v) := -|v-u|_\infty \ . \end{aligned} \tag{30}$$

For measuring the box $[u,v]$ with $b : \mathbb{R}^N \longrightarrow \mathbb{R}_{\geq 0}$, we can use any p-norm with $p \in [1,\infty]$ as well as variants of them

$$\begin{aligned} &b_+^1(y,z,w,u,v) := |\begin{pmatrix} u-\underline{x} \\ v-\overline{x} \end{pmatrix}|_1 \ , \qquad b_+^2(y,z,w,u,v) := \tfrac{1}{2}|\begin{pmatrix} u-\underline{x} \\ v-\overline{x} \end{pmatrix}|_2^2 \ , \\ &b_+^\infty(y,z,w,u,v) := |\begin{pmatrix} u-\underline{x} \\ v-\overline{x} \end{pmatrix}|_\infty \ . \end{aligned} \tag{31}$$

b_- is concave, while b_+ is convex. b_- can be used for unbounded $\boldsymbol{x}$, while this is not possible for b_+. b^2 is smooth, while b^1 and b^∞ are not differentiable. b^1 has an equal growing rate for all components. The growing rate of b^2 depends on $\mathrm{sgn}\big(\frac{1}{2}|.|_2^2 - 1\big)$. b^∞ already grows, if the absolute value of the largest components grows.

Example 5 Choose $n = 1$, $m = 1$, $w = 0$, $c_1 = 1$, $C_1 = \frac{1}{2}$, which yields $F(x) = \frac{1}{2}x^2 + x$ due to (18), as well as $\boldsymbol{x} = [-1,2]$ and consider two CSPs (1) with $\boldsymbol{F}^1 = [-2,1]$ resp. $\boldsymbol{F}^2 = [-2,-1]$ which yield the following two graphics

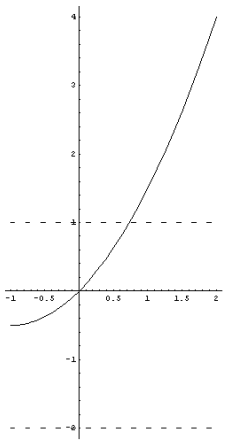


FIG. 9: F for $\boldsymbol{F}^1$

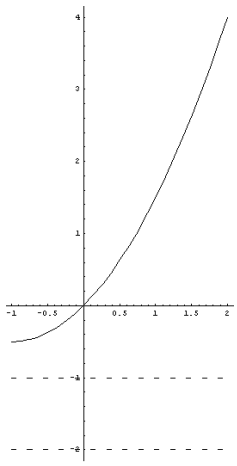


FIG. 10: F for $\boldsymbol{F}^2$

from which we can see that the CSP has feasible points for $\boldsymbol{F}^1$, while it is infeasible for $\boldsymbol{F}^2$. The corresponding certificates f from (15), where we only consider the variables $y \in \mathbb{R}$ and $z \in \boldsymbol{x}$ as well as different T from Example 2, and where we denote the function value of a local minimizer of the optimization problem (27) by $\hat{f}$, can be illustrated by the following plots

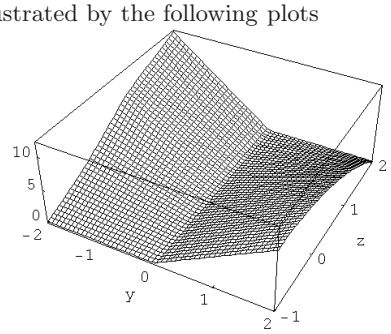


FIG. 11: f for $(\boldsymbol{F}^1, T_1) \Longrightarrow \hat{f} = 0$

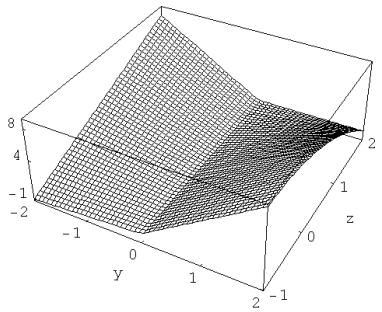


FIG. 12: f for $(\boldsymbol{F}^2, T_1) \Longrightarrow \hat{f} = -1$

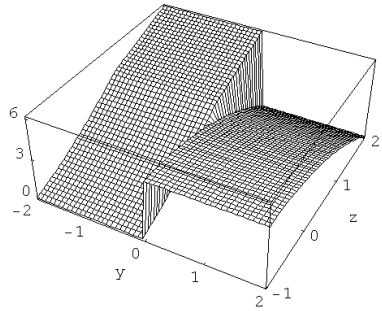


FIG. 13: f for $(\boldsymbol{F}^1, T_2) \Longrightarrow \hat{f} = 0$

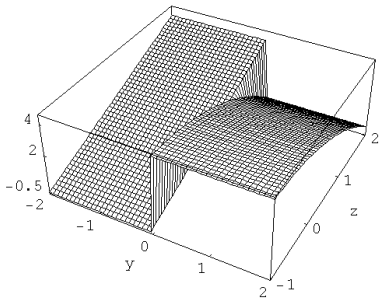


FIG. 14: f for $(\boldsymbol{F}^2, T_2) \Longrightarrow \hat{f} = -\frac{1}{2}$.

We see from Figure 13 and Figure 14 that the certificate f is not defined for $y = 0$ due to the definition of T_2 in Example 2.

3 Starting point

We implemented the suggestions from Subsection 2.4 in GloptLab by Domes [3] which is a configurable MATLAB framework for computing the global solution of a quadratic CSP (1), i.e. with F_k from (18). The matrices $C_k \in \mathbb{R}^{n\times n}$ are lower triangular in GloptLab

$$C_k \in \mathbb{R}^{n\times n}_{\text{tril}} := \{A \in \mathbb{R}^{n\times n} : A_{ij} = 0 \text{ for } i < j\} \ . \tag{32}$$

For running GloptLab the MATLAB toolbox INTLAB by Rump [21], `lp_solve` by Berkelaar et al. [2], SEDUMI by Pólik [20], Sturm [28] as well as SDPT3 by Toh et al. [29] were installed for using all features of GloptLab.

So the last issue that remains to be discussed is, how to find a point (y, z, w, u, v) being feasible for the linearly constrained optimization problem (27) with $f(y, z, w, u, v) < 0$ quickly. For this we need a good starting point $(y^0, z^0, w^0, u^0, v^0)$ and therefore we must take the following observations into account: y^0 and z^0 should be chosen so, that $Y(y^0, z^0)$ is positive, and $(y^0, z^0, w^0, u^0, v^0)$ should be chosen so, that the term $Z(y^0, z^0, w^0, u^0, v^0)$, which is non-negative due to (13), is near zero. These facts lead to the following suggestions for choosing a starting point $(y^0, z^0, w^0, u^0, v^0)$: First of all, if the solver in use can only handle strictly feasible bound/linear constraints (e.g., the second order bundle algorithm by Fendl & Schichl [6] with using `socp` by Lobo et al. [14] for computing the search direction), then the initial choices of u^0 and v^0 must satisfy $u^0, v^0 \in (\underline{x}, \overline{x})$ and $u^0 + r < v^0$, e.g., $u^0 := (1 - t_0)\underline{x} + t_0(\overline{x} - r)$ and $v^0 := (1 - t_1)\underline{x} + t_1(\overline{x} - r)$ for some fixed $t_0, t_1 \in (0, 1)$ with $t_0 < t_1$. Otherwise (e.g., SolvOpt by Kappel & Kuntsevich [11] or the second order bundle algorithm with using MOSEK for computing the search direction) we take the endpoints of $\boldsymbol{x}$ for u^0 and v^0. Secondly, the natural choice for the starting value of $z \in [u, v] \subseteq \boldsymbol{x}$ is the midpoint $z^0 := \frac{1}{2}(u^0 + v^0)$ of the box $[u^0, v^0]$. Thirdly, to get the term $\max\big(0, Y(y, z)\big)$ in the certificate f from (15) as large as possible, we make the following choices: For the case $\underline{F}_k = -\infty$ resp. the case $\overline{F}_k = \infty$ resp. the case that both $\underline{F}_k$ and $\overline{F}_k$ are finite, we choose

$$y_k^0 := \begin{cases} -1 \text{ if } \overline{F}_k < F_k(z^0) \\ 0 \text{ else} \end{cases} , \ y_k^0 := \begin{cases} 1 \text{ if } F_k(z^0) < \underline{F}_k \\ 0 \text{ else} \end{cases} , \ y_k^0 := \begin{cases} 1 \text{ if } \underline{F}_k(z^0) < \underline{F}_k \\ -1 \text{ if } \overline{F}_k < F_k(z^0) \\ 0 \text{ else} \end{cases} \tag{33}$$

respectively due to (16) and (17). Finally, for the choices of R and S we refer to Proposition 6.

Remark 5 If we choose $T = T_2$ (cf. Example 2), then $T(y, z, w, u, v) = 0 \iff y = 0$. Therefore, if $y^0 = 0$ occurs as starting point, then we have a feasible point $F(z) \in \boldsymbol{F}$ due to (33). Furthermore, we can expect that no solver should have difficulties with this choice of T because of the small size of the zero set of T due to Example 2.

In the following we will make use of the MATLAB operators diag, tril and triu.

Proposition 6 *Let F_k be quadratic and let* (32) *be satisfied. Choose any $y \in \mathbb{R}^m$ and consider the modified Cholesky factorization*

$$\hat{A} = \hat{R}^T \hat{R} - D \tag{34}$$

of $\hat{A}$ (with $\hat{R} \in \mathbb{R}^{n\times n}_{\text{triu}}$ and the non-negative diagonal matrix $D \in \mathbb{R}^{n\times n}$), where

$$\hat{A} := C(y) + S^T - S \in \mathbb{R}^{n\times n}_{\text{sym}} \ , \quad S := -\tfrac{1}{2}\text{triu}\big(C(y)^T, 1\big) \in \mathbb{R}^{n\times n}_{\text{striu}} \tag{35}$$

and $\mathbb{R}^{n\times n}_{\text{sym}}$ denotes the space of all symmetric $n \times n$-matrices. Then

$$\hat{A} = C(y) - \tfrac{1}{2}\text{tril}\big(C(y), -1\big) + \tfrac{1}{2}\text{triu}\big(C(y)^T, 1\big) \ , \quad \text{diag}(\hat{A}) = \text{diag}\big(C(y)\big) \ . \tag{36}$$

Furthermore, if we set

$$R := D^{\frac{1}{2}} \ , \tag{37}$$

then $A(y, R, S) = \hat{R}^T\hat{R} \in \mathbb{R}^{n\times n}_{\text{sym}}$.

Proof. Since F_k is quadratic by assumption, the statements of Proposition 3 hold. Since $C(y)$ is lower triangular due to (32) and (20), we obtain $S \in \mathbb{R}^{n\times n}_{\text{striu}}$ and

$$\hat{A} = C(y) - \tfrac{1}{2}\Big(\text{triu}\big(C(y)^T, 1\big)\Big)^T + \tfrac{1}{2}\text{triu}\big(C(y)^T, 1\big) \tag{38}$$

due to (35). Now, (38) implies (36). We calculate

$$C(y)^T - \tfrac{1}{2}\text{triu}\big(C(y)^T, 1\big) = \text{diag}\big(C(y)\big) + \tfrac{1}{2}\text{triu}\big(C(y)^T, 1\big) \ . \tag{39}$$

Because of $\frac{1}{2}\big(\text{triu}\big(C(y)^T, 1\big)\big)^T = \text{tril}\big(C(y), -1\big) - \frac{1}{2}\big(\text{triu}\big(C(y)^T, 1\big)\big)^T$ we have $\text{diag}\big(C(y)\big) + \frac{1}{2}\big(\text{triu}\big(C(y)^T, 1\big)\big)^T = C(y) - \frac{1}{2}\big(\text{triu}\big(C(y)^T, 1\big)\big)^T$. Therefore, combining (38) and (39) yields $\hat{A}^T = \hat{A}$, i.e. $\hat{A} \in \mathbb{R}^{n\times n}_{\text{sym}}$. Consequently there exists a modified Cholesky factorization of $\hat{A}$ of the form (34). Hence, we can choose R according to (37) and evaluating A at (y, R, S) with R from (37) and S from (35) yields $A(y, R, S) = \hat{R}^T\hat{R}$ due to (20), (37) and (34). □

Remark 6 If $\hat{A}$ is positive semidefinite, then $D = 0$ due to (34). If $C(y)$ is a diagonal matrix, then $S = 0$ due to (35). Due to (36), we can construct $\hat{A}$ by setting $\hat{A}$ equal to $C(y)$, then multiplying the lower triangular part of $\hat{A}$ by $\frac{1}{2}$ and finally copying the resulting lower triangular part of $\hat{A}$ to the upper triangular part of $\hat{A}$.

Now we combine the facts that we presented in this subsection to the Algorithm 2, which we will use for creating a starting point for the optimization problem (27) with quadratic F.

Algorithm 2.

```
if the solver can only handle strictly feasible bound/linear constraints
    Choose 0 < t_0 < t_1 < 1 (e.g., t_0 = 0.1 and t_1 = 0.9)
    u = (1 - t_0) x + t_0 (x̄ - r)
    v = (1 - t_1)(x + r) + t_1 x̄
else
    u = x
    v = x̄
end
z = 1/2 (u + v)
F = F(z)
if F ∈ F
    stop (found feasible point => cannot verify infeasibility)
```

```
end if
for k = 1 : m
   if F_k = −∞
      if F̄_k < F_k
         y_k = −1
      else
         y_k = 0
      end if
   else if F̄_k = ∞
      if F_k < F(k)
         y_k = 1
      else
         y_k = 0
      end if
   else
      if F_k < F_k
         y_k = 1
      else if F̄_k < F_k
         y_k = −1
      else
         y_k = 0
      end if
   end if
end for
Compute C(y)
S = −(1/2) triu(C(y)^T, 1)
[R̂, D] = modified_cholesky_factorization(C(y) + S^T − S)
R = sqrt(D)
```

Remark 7 Infeasible constrained solvers (e.g., SolvOpt by Kappel & Kuntsevich [11]) can be applied directly to the nonlinearly constrained optimization problems (28). In this case the starting point created by Algorithm 2 can be used at once without solving optimization problem (27) first as it is necessary for the second order bundle algorithm by Fendl & Schichl [6]. Therefore, the bound constraints $u \leq \hat{u}$, $\hat{v} \leq v$ of optimization problem (28) do not occur in this situation. Nevertheless, it is useful in this case to add the linear constraint $u + r \leq v$ (with a fixed $r > 0$) from optimization problem (27) to the constrained problem for preventing the box $[u, v]$ from becoming too small.

4 Numerical results

In the following section we compare the numerical results of the second order bundle algorithm by Fendl & Schichl [6], MPBNGC by Mäkelä [17] and SolvOpt by Kappel & Kuntsevich [11] for some examples that arise in the context of finding exclusion boxes for a quadratic CSP in GloptLab by Domes [3].

4.1 Introduction

We will make tests for

- (the reduced version of) the second order bundle algorithm for nonsmooth, nonconvex optimization problems with inequality constraints by Fendl & Schichl [6] (with optimality tolerance $\varepsilon := 10^{-5}$ and with MOSEK by Andersen et al. [1] as QCQP-solver for determining the search direction), where we refer to the linearly constrained version as "BNLC" and to the nonlinearly constrained version as "Red(uced) Alg(orithm)". It is an extension of the bundle-Newton method for nonsmooth, nonconvex unconstrained minimization by Lukšan & Vlček [15, 16] to the nonlinearly constrained problems.
- MPBNGC by Mäkelä [17] (with standard termination criterions; since MPBNGC turned out to be very fast with respect to pure solving time for the low dimensional examples in the case of successful termination with a stationary point, the number of iterations and function evaluations was chosen in a way that in the other case the solving times of the different algorithms have approximately at least the same magnitude)
- SolvOpt by Kappel & Kuntsevich [11] (with the standard termination criterions, which are described in Kuntsevich & Kappel [13])

(we choose MPBNGC and SolvOpt for our comparisons, since both are written in a compiled programming language, both are publicly available, and both support nonconvex constraints) on the following examples:

- We give results for the linearly constrained optimization problem (27) with a fixed box (i.e. without optimizing u and v) for dimensions between 4 and 11 in Subsection 4.3.
- We give results for the linearly constrained optimization problem (27) with a variable box (i.e. with optimizing u and v) for dimensions between 8 and 21 in Subsection 4.4.
- We give results for the nonlinearly constrained optimization problem (28) for dimension 8 in Subsection 4.5, where we use $b^1_+(y,z,R,S,u,v) := |\begin{pmatrix} u-\underline{x} \\ v-\overline{x} \end{pmatrix}|_1$ as the objective function.

The underlying data for these nonsmooth optimization problems was extracted from real CSPs that occur in GloptLab by Domes [3]. Apart from u and v, we will concentrate on the optimization of the variables y and z due to the large number of tested examples (cf. Subsection 4.2), and since the additional optimization of R and S did not have much impact on the quality of the results which was discovered in additional empirical observations, where a detailed analysis of these observations goes beyond the scope of this paper. Furthermore, we will make our tests for the two different choices of the function T from Example 2, which occurs in the denominator of the certificate f from (15), where for the latter one f is only defined outside of the zero set of T which has measure zero.

The (extensive) tables corresponding to the results, which we will discuss in this section, can be found in Fendl & Schichl [6].

All test examples will be sorted with respect to the problem dimension (beginning with the smallest). Furthermore, we use analytic derivative information for all occurring functions (Note: Implementing analytic derivative information for the certificate from

(15) effectively, is a nontrivial task) and we perform all tests on an Intel Pentium IV with 3 GHz and 1 GB RAM running Microsoft Windows XP.

We introduce the following notation for the record of the solution process of an algorithm.

Notation 3. *We denote the number of performed iterations by Nit, we denote the final number of evaluations of function dependent data by*

$$\begin{aligned}
\mathrm{Na} &:= \text{“Number of calls to } (f, g, G, F, \hat{g}, \hat{G})\text{”} \text{ (Red Alg)}\\
\mathrm{Nb} &:= \text{“Number of calls to } (f, g, F, \hat{g})\text{”} \text{ (MPBNGC)}\\
\mathrm{Nc} &:= \text{“Number of calls to } (f, F)\text{”} \text{ (SolvOpt)}\\
\mathrm{Ng} &:= \text{“Number of calls to } g\text{”} \text{ (SolvOpt)}\\
\mathrm{N\hat{g}} &:= \text{“Number of calls to } \hat{g}\text{”} \text{ (SolvOpt)} ,
\end{aligned}$$

and we denote the duration of the solution process by

$$\begin{aligned}
\mathrm{t}_1 &:= \text{“Time in milliseconds”}\\
\mathrm{t}_2 &:= \text{“Time in milliseconds (without (QC)QP)”} ,
\end{aligned}$$

where t_2 *is only relevant for the second order bundle algorithm .*

Remark 8 In particular the percentage of the time spent in the (QC)QP in the second order bundle algorithm is given by

$$p_1 := \frac{\mathrm{t}_1(\text{Red Alg}) - \mathrm{t}_2(\text{Red Alg})}{\mathrm{t}_1(\text{Red Alg})} . \tag{40}$$

For comparing the cost of evaluating function dependent data (like e.g., function values, subgradients,. . .) in a preferably fair way (especially for solvers that use different function dependent data), we will make use of the following realistic “credit point system” that an optimal implementation of algorithmic differentiation in backward mode suggests (cf. Griewank & Corliss [9] and Schichl [24, 25, 26]).

Definition 2 Let f_A, g_A and G_A resp. F_A, $\hat{g}_A$ and $\hat{G}_A$ be the number of function values, subgradients and (substitutes of) Hessians of the objective function resp. the constraint that an algorithm A used for solving a nonsmooth optimization problem which may have linear constraints and at most one single nonsmooth nonlinear constraint. Then we define the cost of these evaluations by

$$c(A) := f_A + 3g_A + 3N \cdot G_A + \mathrm{nlc} \cdot (F_A + 3\hat{g}_A + 3N \cdot \hat{G}_A) , \tag{41}$$

where $\mathrm{nlc} = 1$ if the optimization problem has a nonsmooth nonlinear constraint, and $\mathrm{nlc} = 0$ otherwise.

Since the the second order bundle algorithm evaluates f, g, G and F, $\hat{g}$, $\hat{G}$ at every call that computes function dependent data (cf. Fendl & Schichl [6]), we obtain

$$c(\text{Red Alg}) = (1 + \mathrm{nlc}) \cdot \mathrm{Na} \cdot (1 + 3 + 3N) .$$

Since MPBNGC evaluates f, g and F, $\hat{g}$ at every call that computes function dependent data (cf. Mäkelä [17]), the only difference to the second order bundle algorithm with respect to c from (41) is that MPBNGC uses no information of Hessians and hence we obtain

$$c(\text{MPBNGC}) = (1 + \mathrm{nlc}) \cdot \mathrm{Nb} \cdot (1 + 3) .$$

Since SolvOpt evaluates f and F at every call that computes function dependent data and only sometimes g or $\hat{g}$ (cf. Kuntsevich & Kappel [13]), we obtain

$$c(\text{SolvOpt}) = (1 + \text{nlc}) \cdot \text{Nc} + 3(\text{Ng} + \text{nlc} \cdot \text{N}\hat{g}) \ .$$

We will visualize the performance of two algorithms A and B in this section by the following record-plot: In this plot the abscissa is labeled by the name of the test example and the value of the ordinate is given by $\text{rp}(c) := c(B) - c(A)$ (i.e. if $\text{rp}(c) > 0$, then $\text{rp}(c)$ tells us how much better algorithm A is than algorithm B with respect to c for the considered example in absolute numbers; if $\text{rp}(c) < 0$, then $\text{rp}(c)$ quantifies the advantage of algorithm B in comparison to algorithm A; if $\text{rp}(c) = 0$, then both algorithms are equally good with respect to c). The scaling of the plots is chosen in a way that plots that contain the same test examples are comparable (although the plots may have been generated by results from different algorithms).

4.2 Overview of the results

We compare the total time t_1 of the solution process

	t_1(Red Alg)	t_2(Red Alg)	p_1	t_1(MPBNGC)	t_1(SolvOpt)
			$T = 1$		
Linearly constrained (fixed box)	1477	215	0.85	231	2754
Linearly constrained (variable box)	782	60	0.92	30	1546
Nonlinearly constrained	25420	4885	0.81	21860	38761
Nonlinearly constrained (*)	19053	3723	0.80	2067	30312
			$T = \|y\|_2$		
Linearly constrained (fixed box)	1316	129	0.90	15	1508
Linearly constrained (variable box)	797	45	0.94	30	2263
Nonlinearly constrained	24055	4284	0.82	25383	16909
Nonlinearly constrained (*)	18038	3112	0.83	3719	12635

where we make use of (40) and in (*) we consider only those examples for which MPBNGC satisfied one of its termination criterions (cf. Subsection 4.5).

For the linearly constrained problems MPBNGC was the fastest of the tested algorithms, followed by BNLC and SolvOpt. If we consider only those nonlinearly constrained examples for which MPBNGC was able to terminate successfully, MPBNGC was the fastest algorithm again. Considering the competitors, for the nonlinearly constrained problems with $T = 1$ the reduced algorithm is 13.3 seconds resp. 11.3 seconds faster than SolvOpt, while for the nonlinearly constrained problems with $T = |y|_2$ SolvOpt is 7.1 seconds resp. 5.4 seconds faster than the reduced algorithm.

Taking a closer look at p_1 yields the observation that at least 85% of the time is consumed by solving the QP (in the linearly constrained case) resp. at least 80% of the time is consumed by solving the QCQP (in the nonlinearly constrained case), which implies that the difference in the percentage between the QP and the QCQP is small in particular (an investigation of the behavior of the solving time t_1 for higher dimensional problems can be found in Fendl & Schichl [6]).

Therefore, we will concentrate in Subsection 4.3, Subsection 4.4 and Subsection 4.5 on the comparison of qualitative aspects between the second order bundle algorithm, MPBNGC and SolvOpt (like, e.g., the cost c of the evaluations), where before making these detailed comparisons, we give a short overview of them as a reason of clarity of the presentation: In both cases $T = 1$ (solid line) and $T = |y|_2$ (dashed line), where we use the two different line types for a better distinction in the following, we tested 128 linearly constrained examples with a fixed box, 117 linearly constrained examples with a variable box and 201 nonlinearly constrained examples, which yields the following two summary tables consisting of the number of examples for which the second order bundle

algorithm (BNLC resp. the reduced algorithm) is better than MPBNGC resp. SolvOpt (and vice versa) with respect to the cost c of the evaluations

(Color code: Light grey)	MPBNGC					BNLC/Red Alg		
	no termination	significantly better	better	a bit better	nearly equal	a bit better	better	significantly better
					$T = 1$			
Linearly constrained (fixed box)	0	2	5	12	106	2	0	1
Linearly constrained (variable box)	0	0	0	1	116	0	0	0
Nonlinearly constrained	32	6	28	89	31	10	2	3
					$T = \|y\|_2$			
Linearly constrained (fixed box)	0	2	5	30	91	0	0	0
Linearly constrained (variable box)	0	0	0	5	112	0	0	0
Nonlinearly constrained	43	4	28	59	30	15	14	8

(Color code: Black)	SolvOpt					BNLC/Red Alg		
	no termination	significantly better	better	a bit better	nearly equal	a bit better	better	significantly better
					$T = 1$			
Linearly constrained (fixed box)	0	1	3	0	61	25	13	25
Linearly constrained (variable box)	0	0	0	0	48	37	24	8
Nonlinearly constrained	0	0	14	20	21	76	20	50
					$T = \|y\|_2$			
Linearly constrained (fixed box)	0	1	2	1	32	34	49	9
Linearly constrained (variable box)	0	0	0	5	41	32	19	20
Nonlinearly constrained	0	2	24	26	31	61	45	12

that are visualized in Figures 15, 16, and 17

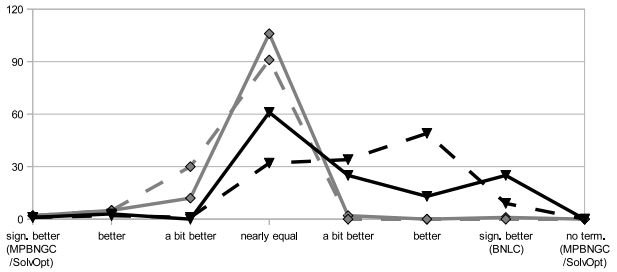


FIG. 15: Linearly constrained with fixed box (summary)

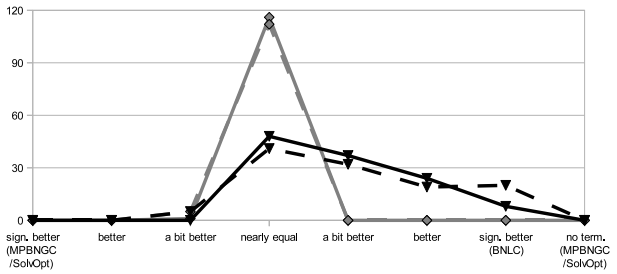


FIG. 16: Linearly constrained with variable box (summary)

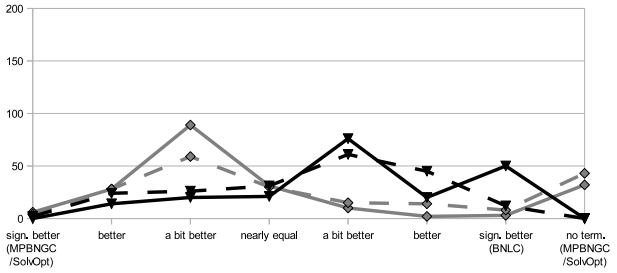


Fig. 17: Nonlinearly constrained (summary)

and that let us draw the following conclusions:

The performance differences between BNLC and MPBNGC can be neglected for the largest part of the linearly constrained examples (with small advantages for MPBNGC in about ten percent of these examples). For the nonlinearly constrained examples the reduced algorithm is superior to MPBNGC in one quarter of the examples, for forty percent of the examples one of these two solvers has small advantages over the other (in most cases MPBNGC is the slightly more successful one), the performance differences between the two algorithms considered can be completely neglected for fifteen percent of the examples, and for further fifteen percent of the examples MPBNGC beats the reduced algorithm clearly.

For the linearly constrained examples BNLC is superior to SolvOpt in one third of the examples, for one quarter of the examples one of these two solvers has small advantages over the other (in nearly all cases BNLC is the slightly more successful one), the performance differences between the two algorithms considered can be completely neglected for forty percent of the examples, and in only one percent of the examples SolvOpt beats the reduced algorithm clearly. For the nonlinearly constrained examples the reduced algorithm is superior to SolvOpt in one third of the examples, for 45 percent of the examples one of these two solvers has small advantages over the other (the reduced algorithm is often the slightly more successful one), the performance differences between the considered two algorithms can be completely neglected for ten percent of the examples, and in the remaining ten percent of the examples SolvOpt beats the reduced algorithm clearly.

In contrast to the linearly constrained case, in which all three solvers terminated successfully for all examples, only the reduced algorithm and SolvOpt were able to attain this goal in the nonlinearly constrained case, too.

4.3 Linearly constrained case (fixed box)

We took 310 examples from real CSPs that occur in GloptLab. We observe that for 79 examples the starting point is feasible for the CSP and for 103 examples the evaluation of the certificate at the starting point identifies the box as infeasible and hence there remain 128 test problems.

BNLC vs. MPBNGC In the case $T = 1$ we conclude from Figure 18 that BNLC is significantly better in 1 example and a bit better in 2 examples in comparison with MPBNGC, while MPBNGC is significantly better in 2 examples, better in 5 examples and a bit better in 12 examples in comparison with BNLC. In the 106 remaining examples the costs of BNLC and MPBNGC are practically the same.

In the case $T = |y|_2$ it follows from Figure 19 that MPBNGC is significantly better in 2 examples, better in 5 examples and a bit better in 30 examples in comparison with BNLC. In the 91 remaining examples the costs of BNLC and MPBNGC are practically the same.

BNLC vs. SolvOpt In the case $T = 1$ we conclude from Figure 20 that BNLC is significantly better in 25 examples, better in 13 examples and a bit better in 25 examples in comparison with SolvOpt, while SolvOpt is significantly better in 1 example and better

in 3 examples in comparison with BNLC. In the 61 remaining examples the costs of BNLC and SolvOpt are practically the same.

In the case $T = |y|_2$ it follows from Figure 21 that BNLC is significantly better in 9 examples, better in 49 examples and a bit better in 34 examples in comparison with SolvOpt, while SolvOpt is significantly better in 1 example, better in 2 examples and a bit better in 1 example in comparison with BNLC. In the 32 remaining examples the costs of BNLC and SolvOpt are practically the same.

4.4 Linearly constrained case (variable box)

We observe that for 80 examples the starting point is feasible for the CSP and for 113 examples the evaluation of the certificate at the starting point identifies the boxes as infeasible and hence there remain 117 test problems of the 310 original examples from GloptLab.

BNLC vs. MPBNGC In the case $T = 1$ we conclude from Figure 22 that MPBNGC is a bit better in 1 example in comparison with BNLC. In the 116 remaining examples the costs of BNLC and MPBNGC are practically the same.

In the case $T = |y|_2$ it follows from Figure 23 that MPBNGC is a bit better in 5 examples in comparison with BNLC. In the 112 remaining examples the costs of BNLC and MPBNGC are practically the same.

BNLC vs. SolvOpt In the case $T = 1$ we conclude from Figure 24 that BNLC is significantly better in 8 examples, better in 24 examples and a bit better in 37 examples in comparison with SolvOpt. In the 48 remaining examples the costs of BNLC and SolvOpt are practically the same.

In the case $T = |y|_2$ it follows from Figure 25 that BNLC is significantly better in 20 examples, better in 19 examples and a bit better in 32 examples in comparison with SolvOpt, while SolvOpt is a bit better in 5 examples (21, 101, 102, 128, 189) in comparison with BNLC. In the 41 remaining examples the costs of BNLC and SolvOpt are practically the same.

4.5 Nonlinearly constrained case

Since we were not able to find a starting point, i.e. an infeasible sub-box, for 109 examples, we exclude them from the following tests for which there remain 201 examples of the 310 original examples from GloptLab.

Reduced algorithm vs. MPBNGC In the case $T = 1$ MPBNGC does not satisfy any of its termination criterions for 32 examples within the given number of iterations and function evaluations. For the remaining 169 examples we conclude from Figure 26 that the reduced algorithm is significantly better in 3 examples, better in 2 examples and a bit better in 10 examples in comparison with MPBNGC, while MPBNGC is significantly better in 6 examples, better in 28 examples and a bit better in 89 examples in comparison with the reduced algorithm, and in 31 examples the costs of the reduced algorithm and MPBNGC are practically the same.

In the case $T = |y|_2$ MPBNGC does not satisfy any of its termination criterions for 43 examples within the given number of iterations and function evaluations. For the remaining 158 examples it follows from Figure 27 that the reduced algorithm is significantly better in 8 examples, better in 14 examples and a bit better in 15 examples in comparison with MPBNGC, while MPBNGC is significantly better in 4 examples, better in 28 examples and a bit better in 59 examples in comparison with the reduced algorithm, and in 30 examples the costs of the reduced algorithm and MPBNGC are practically the same.

Reduced algorithm vs. SolvOpt In the case $T = 1$ we conclude from Figure 28 that the reduced algorithm is significantly better in 50 examples, better in 20 examples and a bit better in 76 examples in comparison with SolvOpt, while SolvOpt is better in 14 examples and a bit better in 20 examples in comparison with the reduced algorithm. In the 21 remaining examples the costs of the reduced algorithm and SolvOpt are practically the same.

In the case $T = |y|_2$ it follows from Figure 29 that the reduced algorithm is significantly better in 12 examples, better in 45 examples and a bit better in 61 examples in comparison with SolvOpt, while SolvOpt is significantly better in 2 examples, better in 24 examples and a bit better in 26 examples in comparison with the reduced algorithm. In the 31 remaining examples the costs of the reduced algorithm and SolvOpt are practically the same.

5 Conclusion

In this paper we presented a nonsmooth function that can be used as a certificate of infeasibility that allows the identification of exclusion boxes during the solution process of a CSP by techniques from nonsmooth optimization: While we can find an exclusion box by solving a linearly constrained nonsmooth optimization problem, the enlargement of an exclusion box can be achieved by solving a nonlinearly constrained nonsmooth optimization problem. Furthermore, we discussed important properties of the certificate as the reduction of scalability and we suggested a method to obtain a good starting point for the nonsmooth optimization problems.

References

1. E.D. Andersen, C. Roos, and T. Terlaky. On implementing a primal-dual interior-point method for conic quadratic optimization. *Mathematical Programming*, B(95): 249–277, 2003.
2. M. Berkelaar, K. Eikland, and P. Notebaert. *lp_solve*. Open source (Mixed-Integer) Linear Programming system (Version 5.1.0.0), May 2004. URL `http://lpsolve.sourceforge.net/`.
3. F. Domes. GloptLab – a configurable framework for the rigorous global solution of quadratic constraint satisfaction problems. *Optimization Methods and Software*, 24(4–5):727–747, 2009. URL `http://www.mat.univie.ac.at/~dferi/gloptlab.html`.

4. H. Fendl. *A feasible second order bundle algorithm for nonsmooth, nonconvex optimization problems with inequality constraints and its application to certificates of infeasibility*. PhD thesis, Universität Wien, 2011.
5. H. Fendl and H. Schichl. A feasible second order bundle algorithm for nonsmooth, nonconvex optimization problems with inequality constraints. In preparation, 2011.
6. H. Fendl and H. Schichl. Numerical results of the feasible second order bundle algorithm for nonsmooth, nonconvex optimization problems with inequality constraints. In preparation, 2011.
7. C.A. Floudas. *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. Oxford University Press, Oxford, 1995.
8. C.A. Floudas. *Deterministic Global Optimization: Theory, Algorithms and Applications*. Kluwer, Dordrecht, 1999.
9. A. Griewank and G.F. Corliss, editors. *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*. SIAM, Philadelphia, PA, 1991.
10. E.R. Hansen. *Global Optimization Using Interval Analysis*. Dekker, New York, 1992.
11. F. Kappel and A.V. Kuntsevich. An implementation of Shor's r-algorithm. *Computational Optimization and Applications*, 15(2):193–205, 2000.
12. R.B. Kearfott. *Rigorous Global Search: Continuous Problems*. Kluwer, Dordrecht, 1996.
13. A.V. Kuntsevich and F. Kappel. *SolvOpt The Solver For Local Nonlinear Optimization Problems*. Karl-Franzens Universität Graz, 1997. URL `http://www.kfunigraz.ac.at/imawww/kuntsevich/solvopt/`.
14. M.S. Lobo, L. Vandenberghe, and S. Boyd. ***socp*** *Software for Second-Order Cone Programming User's Guide*, April 1997. URL `http://stanford.edu/~boyd/old_software/SOCP.html`.
15. L. Lukšan and J. Vlček. PBUN, PNEW – Bundle-Type Algorithms for Nonsmooth Optimization. Technical report 718, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague, Czech Republic, September 1997. URL `http://www.uivt.cas.cz/~luksan/subroutines.html`.
16. L. Lukšan and J. Vlček. A bundle-Newton method for nonsmooth unconstrained minimization. *Mathematical Programming*, 83:373–391, 1998.
17. M.M. Mäkelä. Multiobjective proximal bundle method for nonconvex nonsmooth optimization: FORTRAN subroutine MPBNGC 2.0. Reports of the Department of Mathematical Information Technology, Series B. Scientific computing, B 13/2003 University of Jyväskylä, Jyväskylä, 2003. URL `http://napsu.karmitsa.fi/proxbundle/`.
18. A. Neumaier. *Interval methods for systems of equations*, vol. 37 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 1990.
19. A. Neumaier. Complete search in continuous global optimization and constraint satisfaction. *Acta Numerica*, 13:271–369, 2004.
20. I. Pólik. *Addendum to the SeDuMi user guide version 1.1*, June 2005. URL `http://sedumi.ie.lehigh.edu/`.
21. S.M. Rump. INTLAB - INTerval LABoratory. In Tibor Csendes, editor, *Developments in Reliable Computing*, pp. 77–104. Kluwer Academic Publishers, Dordrecht, 1999. URL `http://www.ti3.tu-harburg.de/~rump/intlab/`.
22. N.V. Sahinidis. BARON. Branch and reduce optimization navigator. User's manual. WWW-Document. URL `http://archimedes.scs.uiuc.edu/baron/baron.html`.

23. N.V. Sahinidis. BARON: A general purpose global optimization software package. *Journal of Global Optimization*, 8(2):201–205, 1996.
24. H. Schichl. The COCONUT environment. Software package. URL `http://www.mat.univie.ac.at/coconut-environment/`.
25. H. Schichl. *Mathematical Modeling and Global Optimization.* Habilitation thesis, Universität Wien, November 2003.
26. H. Schichl. Global optimization in the COCONUT project. *Numerical Software with Result Verification*, pp. 277–293, 2004.
27. L. Schrage. *Optimization Modeling with LINGO.* LINDO Systems, Inc., Chicago, Il, 1999.
28. J.F. Sturm. *Using SeDuMi 1.02, A MATLAB Toolbox for optimization over symmetric cones (Updated for Version 1.05).* Department of Econometrics, Tilburg University, Tilburg, The Netherlads, 1998 – 2001.
29. K.C. Toh, R.H. Tütüncü, and M.J. Todd. *On the implementation and usage of SDPT3 – a MATLAB software package for semidefinite-quadratic-linear programming, version 4.0*, July 2006. Draft. URL `http://www.math.nus.edu.sg/~mattohkc/sdpt3.html`.

A Figures

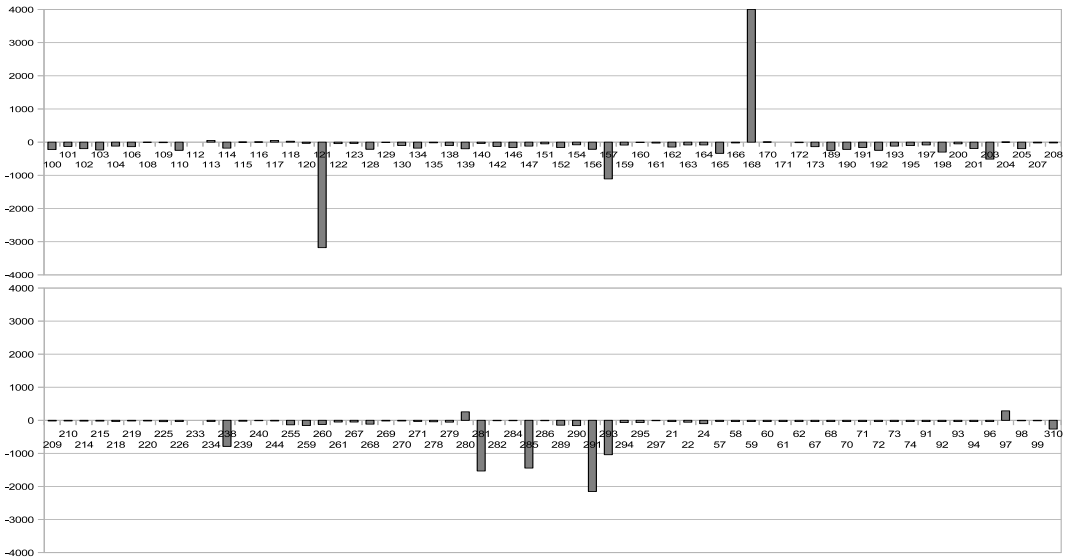


Fig. 18: Linearly constrained (fixed box) — rp(c) for BNLC & MPBNGC ($T = 1$)

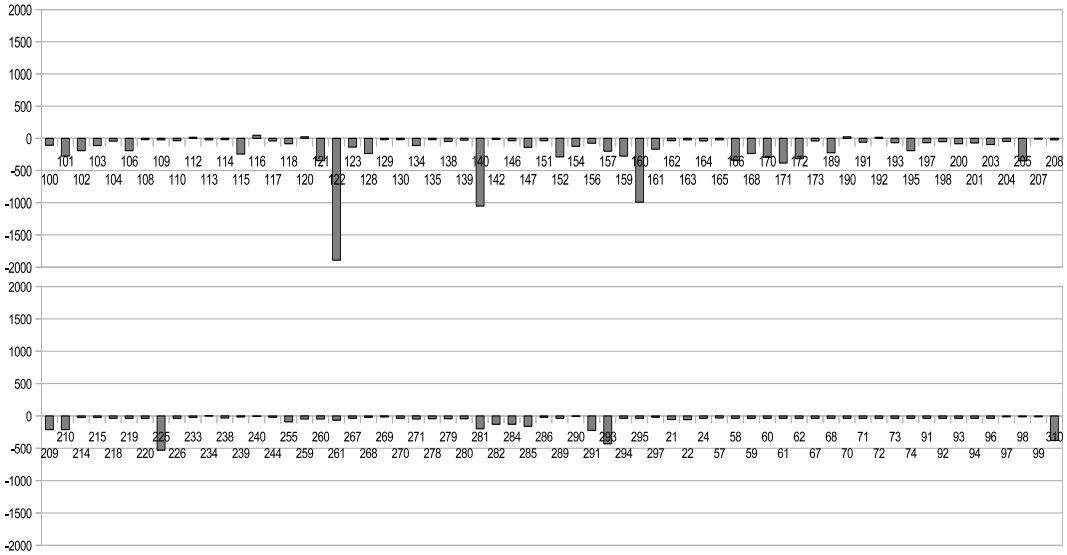


Fig. 19: Linearly constrained (fixed box) — rp(c) for BNLC & MPBNGC ($T = |y|_2$)

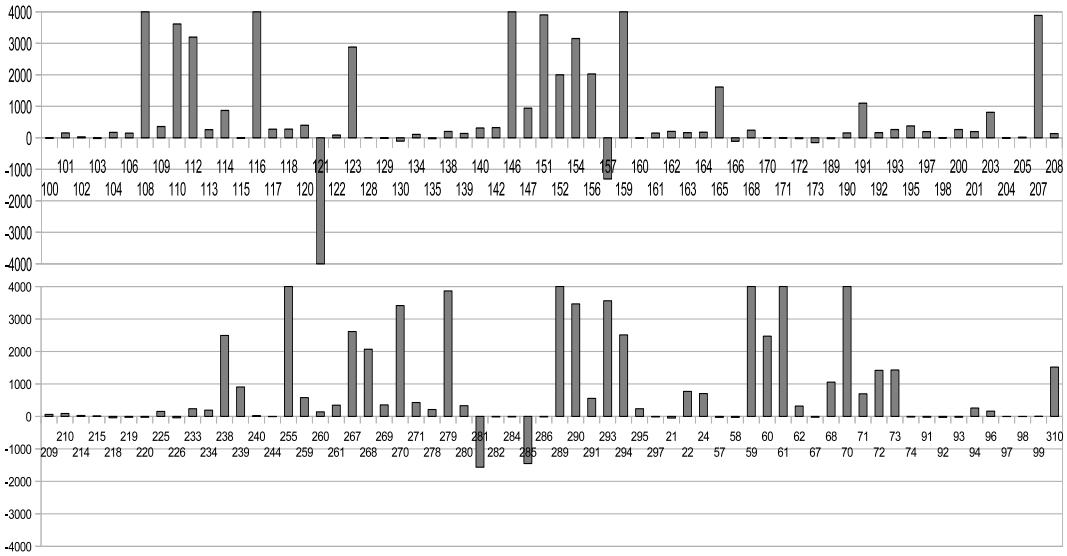


Fig. 20: Linearly constrained (fixed box) — rp(c) for BNLC & SolvOpt ($T = 1$)

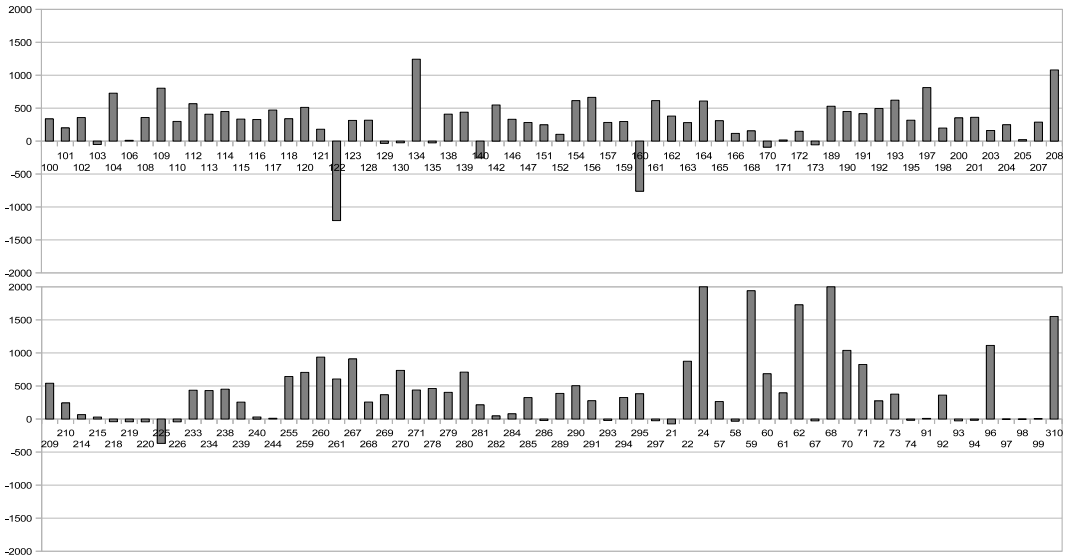


Fig. 21: Linearly constrained (fixed box) — rp(c) for BNLC & SolvOpt ($T = |y|_2$)

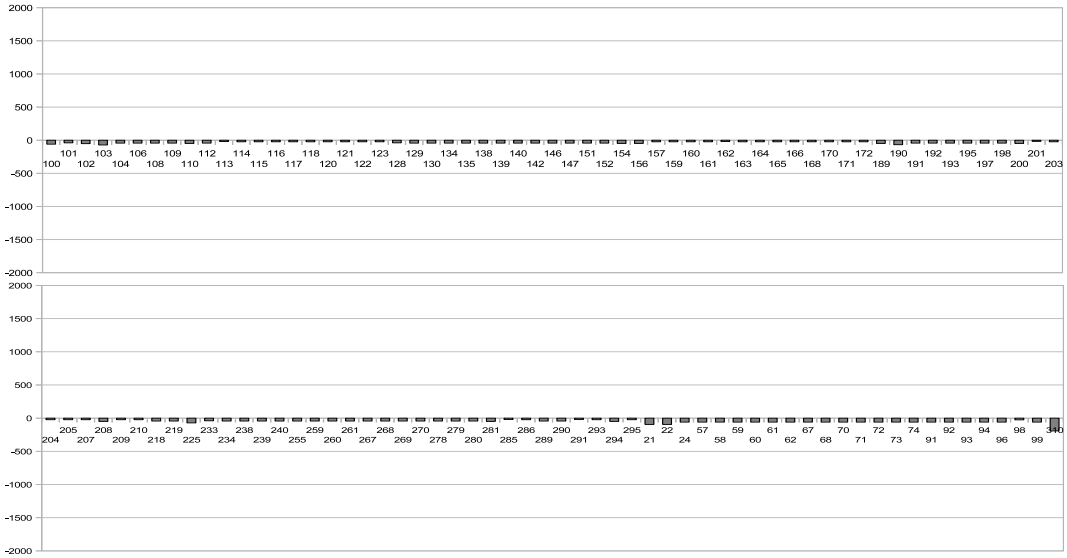


Fig. 22: Linearly constrained (variable box) — rp(c) for BNLC & MPBNGC ($T = 1$)

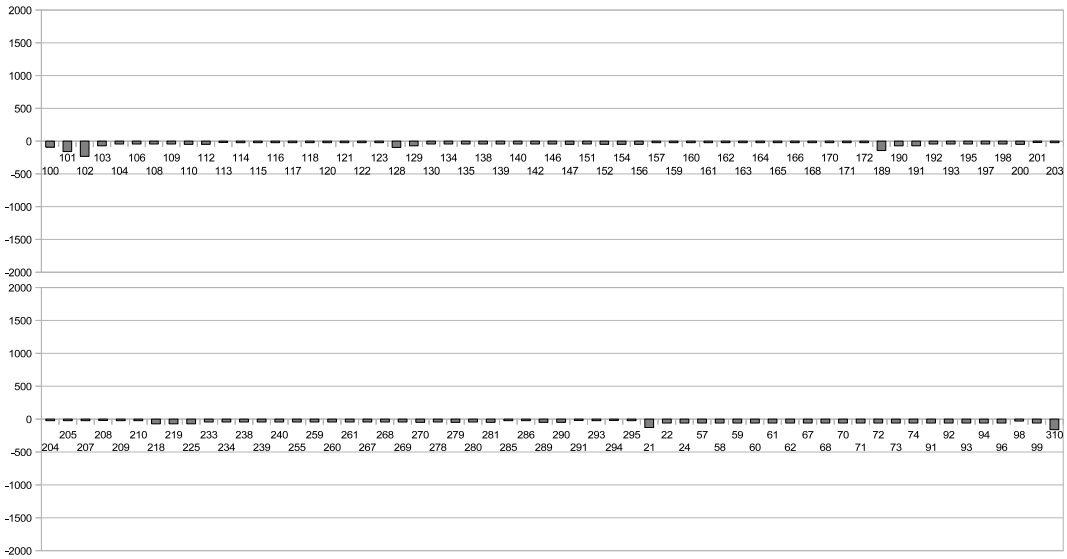


FIG. 23: Linearly constrained (variable box) — rp(c) for BNLC & MPBNC ($T = |y|_2$)

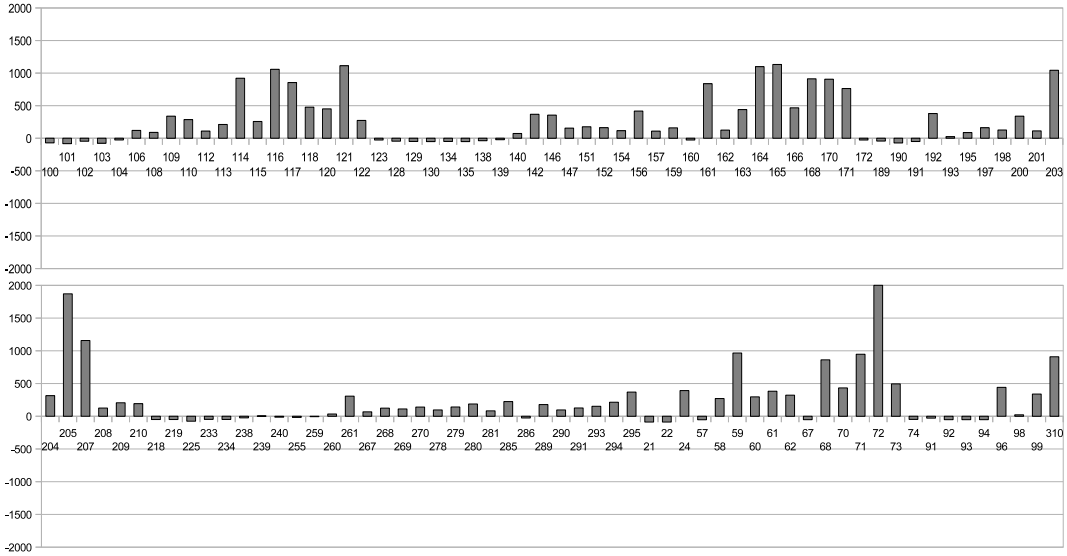


FIG. 24: Linearly constrained (variable box) — rp(c) for BNLC & SolvOpt ($T = 1$)

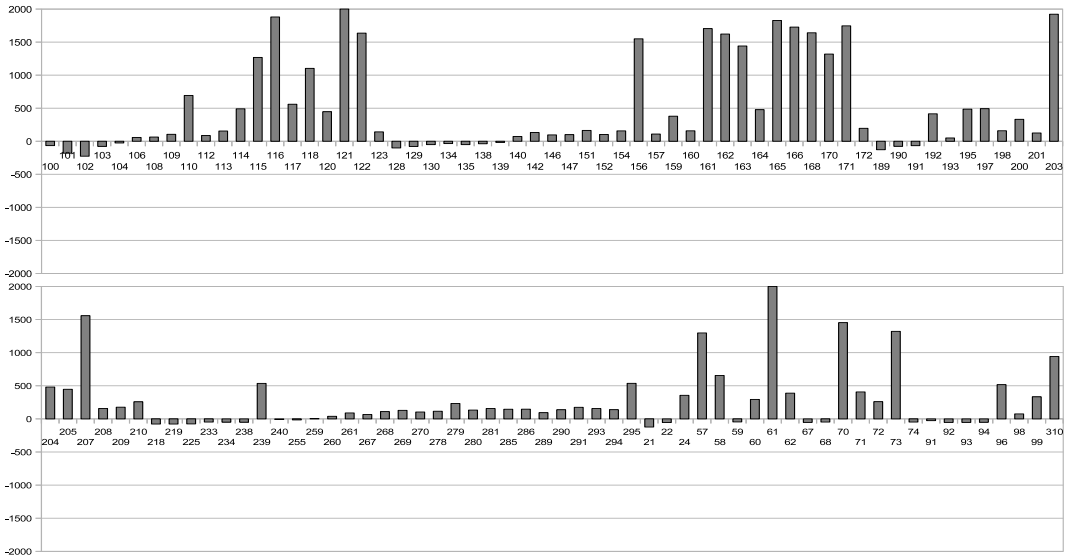


FIG. 25: Linearly constrained (variable box) — rp(c) for BNLC & SolvOpt ($T = |y|_2$)

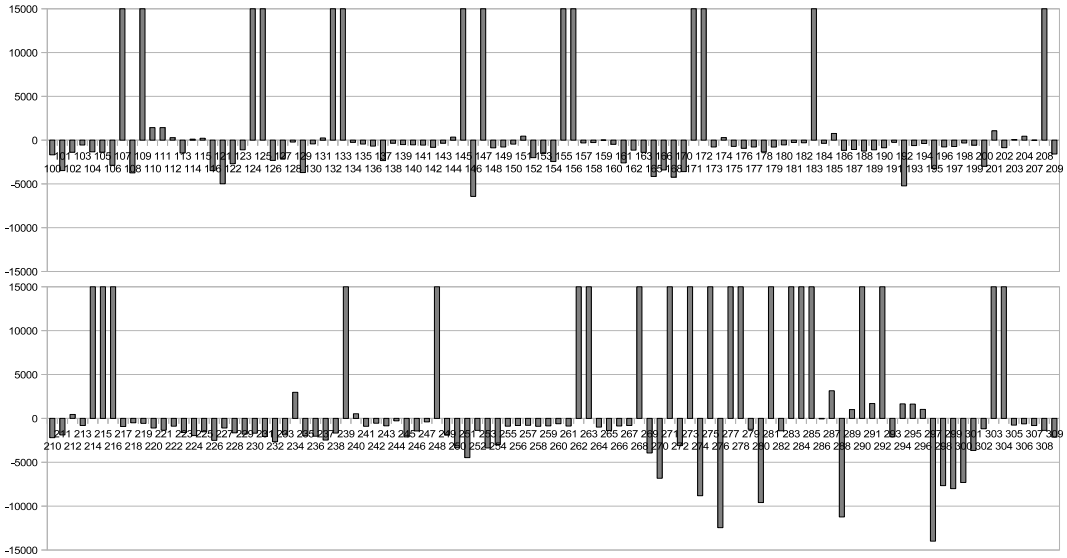


FIG. 26: Nonlinearly constrained — rp(c) for Red Alg & MPBNGC ($T = 1$)

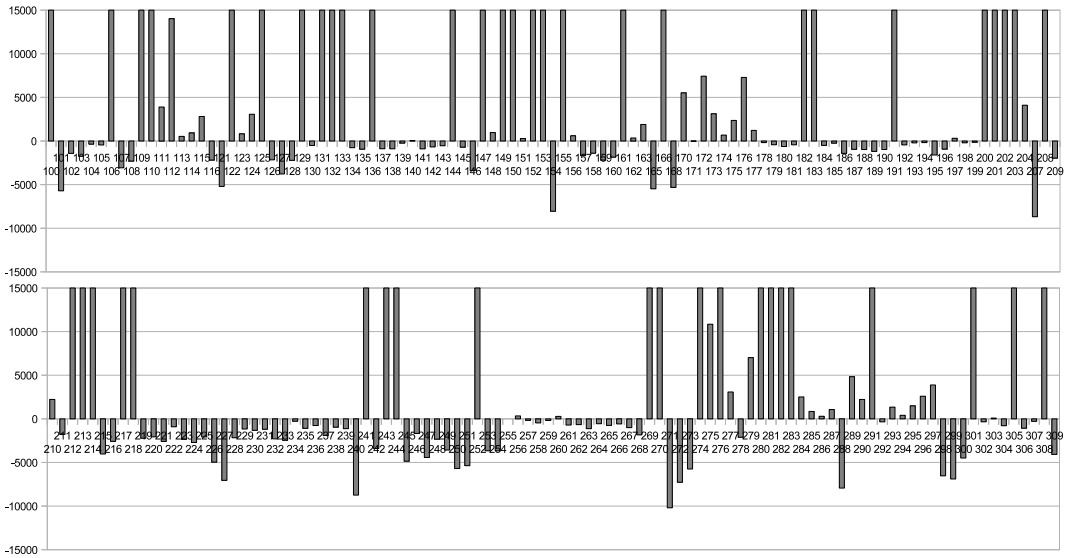


FIG. 27: Nonlinearly constrained — rp(c) for Red Alg & MPBNGC ($T = |y|_2$)

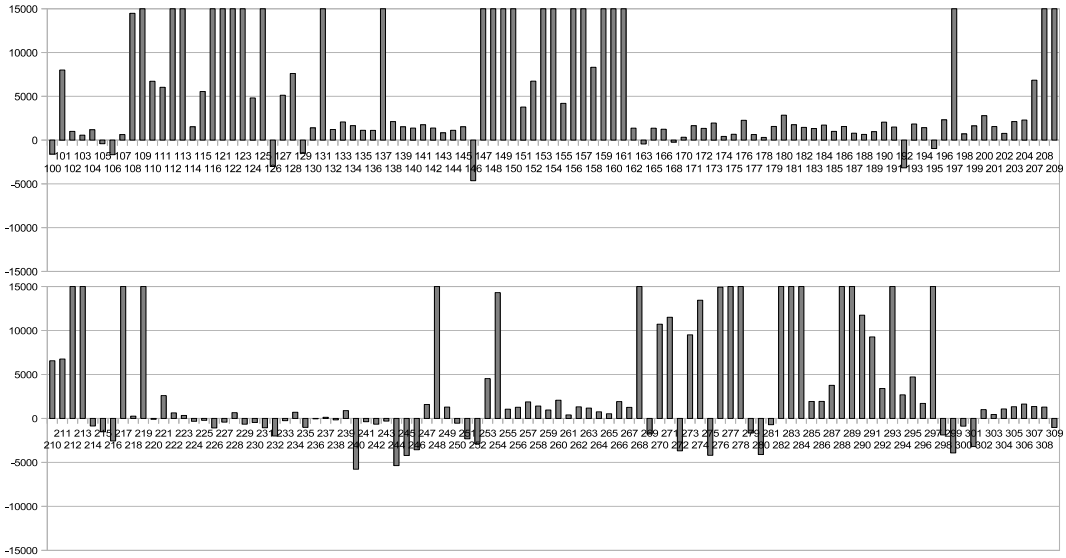


FIG. 28: Nonlinearly constrained — rp(c) for Red Alg & SolvOpt ($T = 1$)

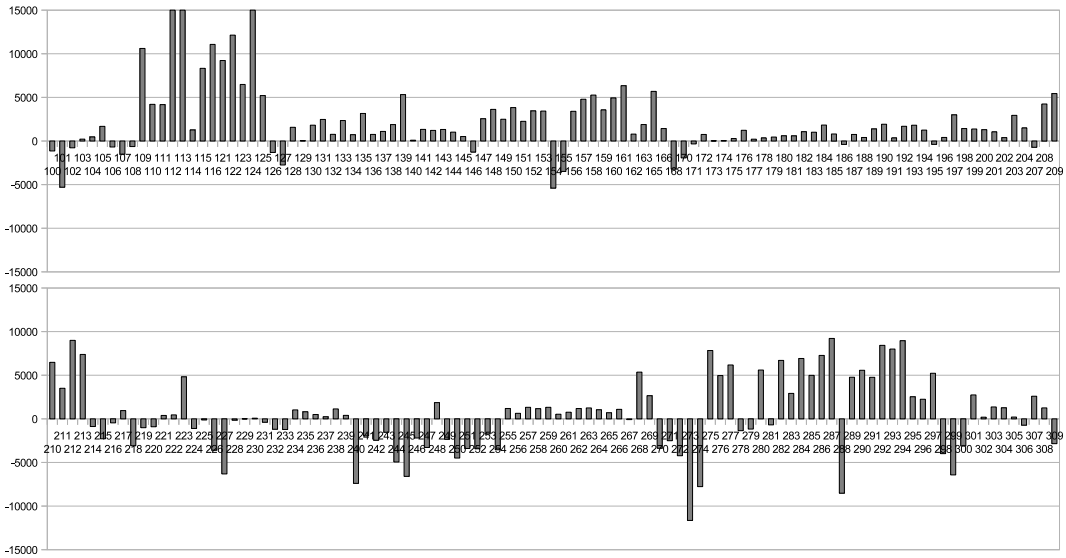
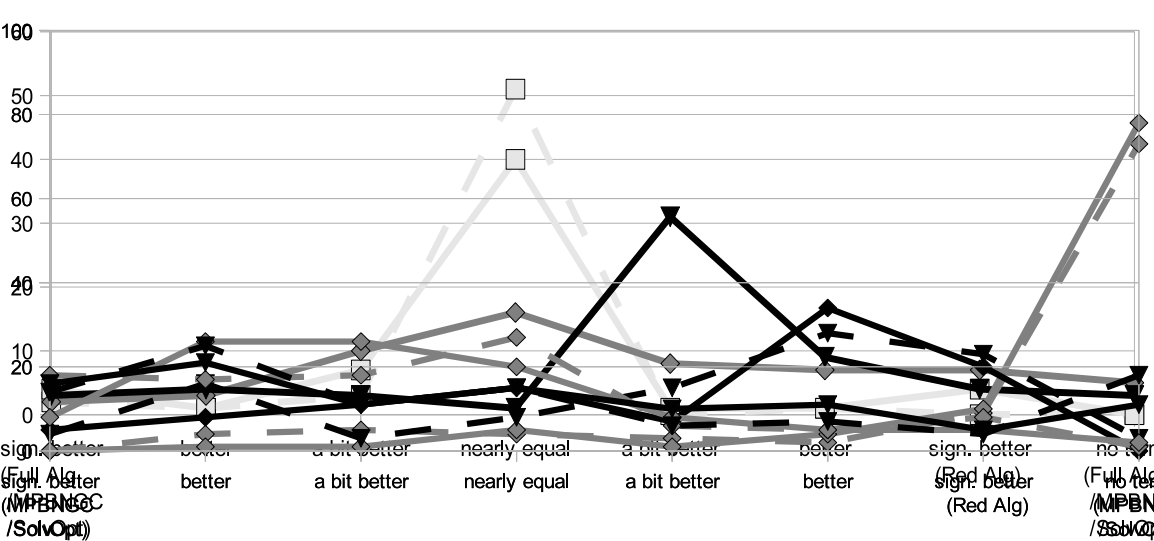


Fig. 29: Nonlinearly constrained — rp(c) for Red Alg & SolvOpt ($T = |y|_2$)


100
50
80
40
60
30
20
10
0
better
a bit better
nearly equal
a bit better
better
sign. better
(Red Alg)
no sign.
(Full Alg
/MPBNGC
/SolvOpt)